A framework for monitoring SIP enterprise networks

Mohamed Nassar, Radu State, Olivier Festor

A framework for monitoring SIP enterprise networks

Mohamed Nassar, Radu State, Olivier Festor
INRIA research center, Nancy - Grand Est
615, rue du jardin botanique, 54602
Villers-Lès-Nancy, France
Firstname.Lastname@loria.fr

***Abstract*—In this paper we aim to enable security within SIP enterprise domains by providing monitoring capabilities at three levels: the network traffic, the server logs and the billing records. We propose an anomaly detection approach based on appropriate feature extraction and one-class Support Vector Machines (SVM). We propose methods for anomaly/attack type classification and attack source identification. Our approach is validated through experiments on a controlled test-bed using a customized normal traffic generation model and synthesized attacks. The results show promising performances in terms of accuracy, efficiency and usability.**

I. Introduction

SIP (Session Initiation Protocol - RFC 3261 [1]) is the defacto signaling protocol for Internet and Internet Multimedia Subsystems (IMS). SIP supports a new range of IP multimedia applications (e.g. Voice over IP (VoIP), IP TeleVision (IPTV), Video on Demand (VoD)) having new requirements in terms of security. In terms of availability and quality of service, these applications need to be protected against Denial of Service (DoS) and flooding attacks. The service providers may lose their subscribers because of annoyance attacks such as Spam over Internet Telephony (SPIT). They may lose costly financial revenues within a short period of time because of successful fraud attempts. Other threats include Caller-Id spoofing, user profiling, eavesdropping and authentication break-in. The critical aspects of these services impose a real-time and continuous monitoring in order to detect anomalies and intrusions. The monitoring of SIP networks is challenging because of their open, dynamic and large scale properties. The complexity of the SIP infrastructures rise the cost of security operations like maintenance and upgrading. Fast diagnostic and automated response/recovery mechanisms are required in order to limit the impact of successful attacks on the supported services.

In this context, we propose an overall monitoring framework composed of three phases: the monitoring phase, the detection phase and the correlation phase. The monitoring phase is based on three important data sources that are typically available: (1) The network traffic (especially the SIP traffic), (2) the SIP servers statistics and logs, and (3) the Call Detail Records (CDRs). We define a set of relevant features to characterize each of these data sources. The monitoring phase is followed by a detection phase where Support Vector Machines (SVM) [2] learning techniques are applied. SVM are a relatively recent kernel-based learning method that is known for its advanced ability processing high dimensional data. The correlation phase is rule-based aimed at interpreting the detection results across the three data sources. We strive for our framework three essential properties: (1) the detection accuracy, (2) the real-time efficiency and (3) the usability in real world settings. Revealing the evidence of an abnormal situation is thus not sufficient. We have to answer the following questions:

- What is the type of the anomaly? Is it a configuration problem of some device? or an attack of a specific type?
- What is the intensity of the attack? Is it tolerable? or should we act immediately in order to prevent it?
- What is the attack source(s)? At which level should we act and block it?

We also propose a traffic generation model for emulating the behavior of real subscribers. We demonstrate the deployment of our framework within an illustrative SIP network. We validate our approach through experiments on one data source (the server statistics) and show the performances of the detection phase. We address the feature selection and visualization by using a method for the estimation of the prediction sensitivity of each feature.

The rest of this paper is organized as follow: In the next section we provide background information about the SIP protocol. Section II describes the monitoring phase. The SVM model is depicted in section IV. Section V exposes the experiments and the results. The related works are discussed in Section VI. Finally Section VII concludes the paper and mentions the future works.

II. SIP background Information

Basically, SIP allows two or more communicating parties to set up, modify and terminate a media session. The media session is described through the Session Description Protocol (SDP) body carried by the SIP messages. SIP is text-based with heritage from HTTP and SMTP. A SIP message starts by a request line and contains several headers for routing purposes (e.g. From, To, Via, Contact, Record-route and Route). SIP defines 13 request types (or methods) and 6 families of responses. The most important requests are: INVITE (to request establishing a media session),

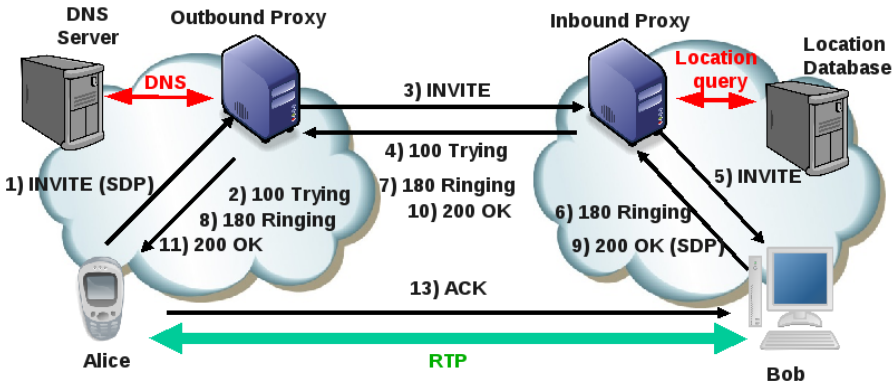

Figure 1. Example of a successful SIP call establishment

REGISTER (to register the location of a user agent at a SIP registrar), BYE (to terminate an established media session), ACK (to acknowledge the final response to an INVITE request) and CANCEL (to terminate a pending call attempt). The families of responses are: Informational (1xx e.g. 180 Ringing), Success (2xx), Redirection (3xx), Client Error (4xx), Server Error (5xx) and Global Error (6xx). A SIP transaction is composed of a request and one or more responses. A SIP Dialog is composed of one or more transactions. There are four types of transactions: on the one hand client transactions and server transactions, on the other hand INVITE transactions and non-INVITE transactions. A final response is acknowledged by an ACK message only in the case of an INVITE transaction. An ACK message in response to a success final response (200 OK) is considered as a separate transaction. The SIP addressing scheme is based on the Uniform Resource Identifier (URI) e.g. `sip:user@host:port;parameters`. The SIP call establishment is based on the so-called the SIP trapezoid. An example of a successful establishment is shown in Figure 1. The SIP messages are numbered in the order of their appearance in the time.

SIP supports many security mechanisms that can be used hop-by-hop (e.g. IPsec, TLS) or end-to-end (S/MIME certificates, Secure SIP URI scheme (SIPS)). While some SIP headers can not be encrypted because they are needed by intermediary proxies for routing, encryption of SDP bodies prevents eavesdropping and media injection attacks. The authentication of REGISTER and INVITE messages are based on an HTTP-like challenge scheme. An enterprise SIP-based network is composed of several functional entities: User Agents (UA) generate or terminate media sessions and maintain the state of calls. Presence agents notify their subscribers about presence information that has been published by other entities. Back-to-Back User Agents (B2BUA) behave at the name of other UAs and hide their identities by declaring themselves as the origin of the requests. Proxy servers help routing the SIP messages while preserving the end to end transparency between the sender and the receiver. There is tow types of proxy servers: stateful and stateless. The difference is that stateful proxy keep the state of ongoing transactions and retransmit messages when lost. Redirect servers do not route messages but instead provide location information. Registrar servers create temporary binding between the Address of Record (AoR) (e.g. the phone number) and the IP location of the registering device. Non-specific SIP entities such as gateways and PBXs are also needed in order to interconnect with other networks and applications.

III. The Monitoring Phase

The monitoring plan relies on defining a relevant set of probes (or features) and auditing/computing them periodically in order to profile a data source. Feature extraction is a critical step in intrusion detection since it aims at determining what evidence that can be taken from raw audit data is most useful for analysis. We consider three important data sources in a typical SIP/VoIP network: the SIP traffic, the servers' logs and statistics, and the billing records.

A. The SIP traffic

The SIP traffic is the typical data source for application-specific network-based anomaly detection. The extraction of features from raw SIP traffic imposes a certain cost in terms of computation delay and memory consumption. Using a certain cost-level of features is constrained by the desired real-time efficiency of the detection system. We can categorize the SIP traffic features into 3 cost levels:

- message-based features: can be computed directly upon the receiving of a SIP message (e.g. request distribution, response distribution, inter-arrival times).
- transaction-based features: require rebuilding of all the transactions from the SIP traffic (e.g. distribution of final state of INVITE transactions, non-INVITE transactions, client transactions and server transactions). A memory entry is reserved for each transaction that is

alive. Since the lifetime of transactions is relatively small, these features are not very costly in terms of memory.

- dialog-based features: require rebuilding of all the dialogs from the SIP traffic (e.g. distribution of final state of sessions). A memory entry is reserved for each dialog that is alive. These features can be very costly in terms of memory since some dialogs have a relatively high lifetime (in the order of hours). Using persistent storage solves this problem while inducing an additional read/write access time.

Moreover, since the good functioning of SIP relies on the interaction with other protocols such as DNS, Radius, RTP or MGCP defining statistics on these interactions (e.g. the number of DNS requests per SIP session) is important for the detection of cross-protocol attacks (e.g. flooding based on an invalid domain name).

B. The servers' statistics and logs

The logs of the SIP servers and the statistics provided by their management and monitoring interfaces are the typical data source for host-based intrusion detection. Attributing monitoring probes is dependent on the software design and the capabilities of each server. We study the case of the SER-like proxy servers[1] which are known for their reliability and high performance. The design of a SER-like proxy involve a core and several modules as shown in Figure 2. The interpreter helps configuring the routing logic as required by the administrator. The SIP parsing and transport entities are used by the routing logic in order to properly process and forward the SIP messages. The core interacts with the proxy modules through dedicated interfaces. The modules represent different applications, database back-ends and management interfaces. Thanks to this modular architecture, the management interface provides statistics about the core and the different modules used by the proxy. We particularly experimented with the Opensips server and we based our monitoring scheme on the statistics shown in Table I.

These statistics or features can be audited periodically in order to profile the internal state of the SIP proxy in real-time. In terms of accuracy requirements, the features must be proven to have values in normal situations that are significantly different than values in abnormal or attack situations. For example, the memory statistics of SER-like servers does not fulfill this condition. In fact, some attacks cause a change in the memory statistics' values that persists even after their disappearance. We therefore exclude these statistics from the monitoring plan.

[1]Ser, Openser, Opensips and Kamilio have the same design as they are all offshoots of the former SER project

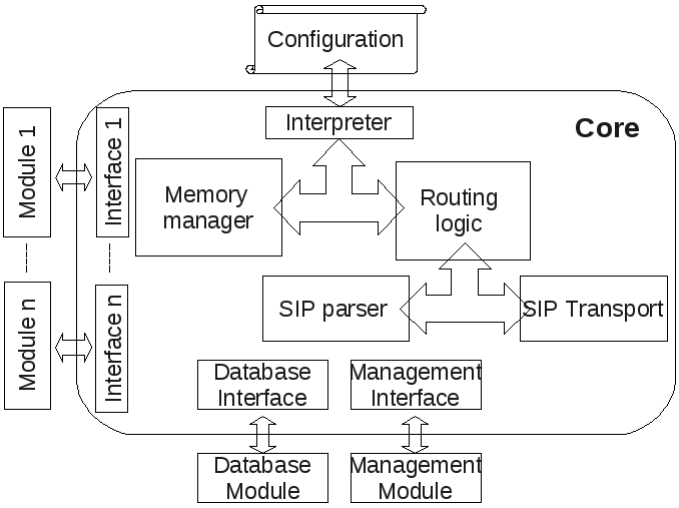

Figure 2. SER-like Proxy design

C. The call activities-Accounting

The Call Detail Records (CDRs) are an important source of information for profiling users, group of users and global call activities. Fraud scenarios and compromised nodes cause abnormal call behaviors. We profile a group of CDRs through the following features:

- The call rejection rate: is the percentage of calls that have their end status as "Failed" or "Busy".
- The call recipient rate: is the ratio of distinct call recipient to the overall number of calls.
- The distinct callers rate: is the ratio of distinct callers to the overall number of calls.
- The average Billing duration: The billing duration is the time between the 200 OK and the BYE message in the successful call signaling.
- The average Call duration: The call duration is the time between the INVITE and the BYE message.
- The call rate: is the frequency of calls within a period of time.
- The target application rate: The target application is the last application performed during the call (e.g. forwarding to the callee, music on hold, voicemail, etc.).
- The context rate: The context of a call reveals the class of the dialed extension (e.g. incoming/outgoing, local/department/international).

This set of features is general and has to adapted to each level. For example, the distinct call rate is excluded at the user level. The average call duration should be extended at the global level by defining a distribution over several intervals of call duration. We suggest building several call profiles depending on context parameters (e.g. weekday vs. weekend day, calls from home vs. calls from work, etc.).

Table I
OPENSIPS FEATURES

Group	Feature	Description	Group	Feature	Description
Core	1	received requests		26	User Agent Server transactions
statistics	2	received replies		27	User Agent Client transactions
	3	forwarded requests		28	2xx transactions
	4	forwarded replies		29	3xx transactions
	5	dropped requests		30	4xx transactions
	6	dropped replies		31	5xx transactions
	7	erroneous requests		32	6xx transactions
	8	erroneous replies		33	in-use transactions
	9	invalid URI received	User Location	34	registered users
	10	unsupported SIP methods	statistics	35	location users
	11	bad message headers		36	location contacts
Net statistics	12	UDP packets waiting to be read		37	location expires
	13	TCP packets waiting to be read	Registrar	38	maximum expires
Stateless	14	1xx replies	statistics	39	maximum contacts
statistics	15	2xx replies		40	default expires
	16	3xx replies		41	accepted registrations
	17	4xx replies		42	rejected registrations
	18	5xx replies	URI statistics	43	positive checks
	19	6xx replies		44	negative checks
	20	sent replies	Dialog	45	active dialogs
	21	sent error replies	statistics	46	early dialogs
	22	received ACKs		47	processed dialogs
Transaction	23	received replies		48	expired dialogs
Manager	24	relayed replies		49	failed dialogs
statistics	25	local replies			

IV. THE SVM DETECTION PHASE

Support Vector Machines (SVM) [2] are known for their efficiency and accuracy in many application domains namely network-based and payload-based anomaly detection. SVM are also lightweight hence suitable for a run-time monitoring scheme. One-class SVM constitutes a geometric framework where the statistics are mapped into a feature space and anomalies are detected in sparsely populated regions. One-class SVM are particularly suitable for anomaly detection (when only normal data are available).

A. Background on one-class SVM

The basic idea of one-class SVM is to separate the points from the origin with the largest possible margin by means of a hyperplane. Alternatively, The hypersphere formulation suggests rather finding the smallest sphere enclosing the data points. The quarter-sphere formulation [3] is a special case of the sphere SVM which is more adapted to one-sided data on $\mathbb{R}_0^+$.

Solving the one-class SVM problem leads to a decision function that helps predicting the class of unseen data. Usually the sign of the decision function decides if a data sample belongs to the normal class (if the sign is positive) or to the abnormal class (if the sign is negative). The idea of the quarter-sphere SVM is to tune the point of decision in order to ameliorate the accuracy (i.e. tuning the radius of the sphere). That is, the anomaly score of a point is the value of the decision function at this point (i.e. the distance from this point to the center of the sphere).

Formally, given training vectors $\overrightarrow{x_i} \in R^n, 0 < i \leq l$ and no class information, the problem of finding the separating hyperplane with maximum margin from the origin is stated herein:

$$\text{Find } \overrightarrow{w}, \text{ b, } \xi \text{ and } \rho \text{ to minimize } \frac{1}{2}\overrightarrow{w}.\overrightarrow{w} - \rho + \frac{1}{\nu l}\sum_i \xi_i$$
$$\text{Subject to } \begin{cases} \overrightarrow{w}.\overrightarrow{x_i} \geq \rho - \xi_i, \forall \overrightarrow{x_i} \\ \xi_i \geq 0, \forall i \end{cases}$$

The slack variables ξ_i are used to allow a certain degree of misclassification. The ν parameter is an upper bound over the fraction of the data points that can be put outside the boundary. $\overrightarrow{w}$ and b define the separating hyperplane. The $\rho < 1$ parameter (so-called soft-margin) is used to obtain the desirable ratio of support vectors. The support vectors are the data points having non-zero slack variables (geometrically are lying on the margin).

The non-linear separation replaces scalar products using a kernel function. The goal of the kernel function is to map the data into a high dimensional space where the optimal separator is determined. One of the most known kernel functions is the Radial Basis Function (RBF) kernel:

$$k_{rbf}(\overrightarrow{x}, \overrightarrow{z}) = exp(-\gamma|\overrightarrow{x} - \overrightarrow{z}|^2)$$

where $\gamma > 0$.

B. Anomaly type identification

Once an anomaly is detected, a response can not be triggered if no information about the anomaly is provided. We suggest three methods for the attack identification:

- If labeled data of attack types are available, SVM multi-classification techniques can be used (e.g. C-SVC).

Another idea is to represent each anomaly type by a one-class SVM so that the data instance can be checked if it belongs to it or not.

- Each attack type has qualitative properties. Multiple attacks can be classified on a rule-basis using a decision tree. That is, when an anomaly is detected the values of the different features are checked by the decision rules.
- Using specific visualization techniques such as the prediction sensitivity for quarter-sphere SVM [4]. The prediction sensitivity measures the degree to which prediction is affected by adding weight to a particular feature. This method determines the most relevant features to the detection hence gives insight about the attack type.

C. Anomaly source identification

It is also important to determine the attack source when an anomaly is detected at a global level. We mean by the global level that several call sources contribute to the monitored data source or features. For example, the average call duration of a group of users can be decomposed to the average call duration for each of the users. We suggest that the attack source identification can be based on this fact: the suppression of the attack source effects from the computation (when possible) moves the anomaly point to lie again in the normal region. The anomaly score is decomposed to individual scores corresponding to the participating sources, then we apply the following algorithm:

1) range all the call sources by decreasing order of anomaly score,
2) suppress the top most ranked source and add it to the list of suspicious sources,
3) recalculate the data point,
4) test the data point to fall into the normal region,
5) if the new data point is predicted as normal then return the list of suspicious sources and exit, otherwise come back to the first step with the remaining call sources.

Once the anomaly source is identified, an appropriate response can be triggered without harming the availability of the service (for example, resetting the password of a subscriber or throttling calls from a certain VoIP provider).

V. Experimentation and Results

The experiments are run over a controlled test-bed using synthesized normal traffic and a set of attack tools. We briefly describe the traffic generation model and the test-bed then we discuss the detection results.

A. Traffic generation model

The traffic generation model is based on defining profiles of emulated SIP user agents so-called VoIP bots which are initially designed as proof of concept attack tools [5]. The bots are remotely managed through Internet Relay Chat (IRC) protocol. The goal of using IRC is to ease the configuration and the control of a large number of bots. At booting, each bot connects to a predefined room (channel) on a local IRC server. Each bot registers to one or more registrars in order to send and receive calls. Emitting calls follows a Poisson distribution where the mean number of calls per unit of time ($\lambda_{Poisson}$) is set. The call holding time follows an exponential distribution where the mean of the call holding time is set ($1/\lambda_{Exp}$). The bots generate calls to peers that are socially-connected to them with a probability P_{social} and to other peers with a probability $1-P_{social}$. The social connections are formed by randomly repartitioning the bots among several inter-connected cliques. The bots take the first received call and responds with "busy" to the following calls as long as they are "in-call". The ringing time follows a uniform distribution with relatively small durations. In order to limit the time of experiments, we configured the bots with relatively high $\lambda_{Poisson}$ (e.g. 100 calls/hour) and relatively low $1/\lambda_{Exp}$ (e.g. 1-10 seconds). In order to limit the resource utilization, we limit the number of deployed bots per machine to 50.

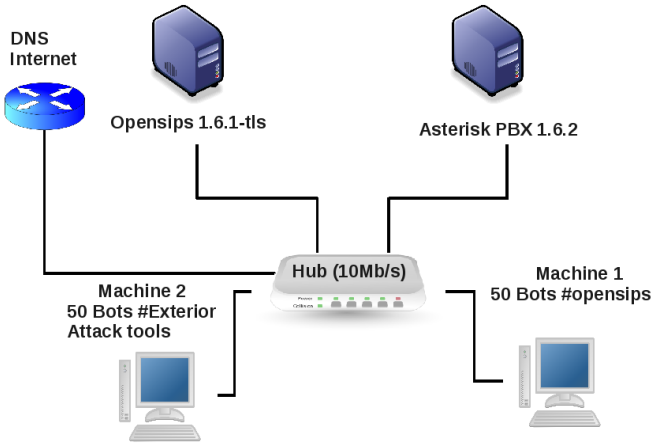

Figure 3. VoIP Test-bed

An attack (or anomaly) is mainly described by the attack tool, the destination URI(s), the source, the target and the intensity. Several SIP attack tools are available like Inviteflood, Sipscan and Spitter[2]. We make use of the Inviteflood tool to simulate a flooding attack. The tool targets the victim with INVITE messages without acknowledging any response causing it to retransmit the signaling many times. The SIP servers have different behavior depending on the destination URI of the attack (e.g. Invalid username, Invalid IP, Invalid domain name).

B. Test-bed

As shown in Figure 3, the test-bed contains open-source SIP servers (Opensips[3] 1.6.1 and Asterisk PBX[4] 1.6.2), 2

[2]http://www.hackingvoip.com/sec_tools.html
[3]http://www.opensips.org/
[4]http://www.asterisk.org/

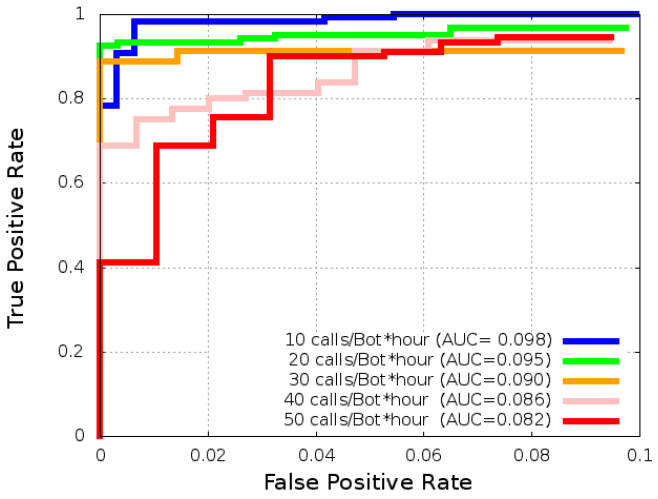

Figure 4. Detection accuracy under different background traffic volumes

groups of bots representing Opensips users and external domains, and the attack tools. The Asterisk has the Opensips server as an outgoing proxy. We emulate this situation by configuring the Spitter tool over Asterisk and generate calls towards Opensips. Routing the calls among the bots is fixed at the Opensips routing logic. Physically, the test-bed is composed of 4 machines (Intel Pentium 4 CPU 3.40GHz and 2G RAM memory running Ubuntu 9.10) connected through a hub. The DNS and the external connection are provided by a router and an address translator.

The bots connect to two channels (rooms) at the IRC server (#opensips and #exterior) and wait for commands. We start each experiment by asking the bots to register using their identifiers, than to start issuing calls following the call generation model parameters. Using this test-bed, we generate several data-sets corresponding to different traffic background volumes and different kind of anomalies.

C. Detection results

The detection accuracy depends on SVM-specific parameters like the used kernel and the training anomaly factor (ν), the size (and the cleanness) of the training data-set and monitoring-specific parameters like the monitoring cycle (in time unit) and the training time. Experiments on only one data source (the Opensips servers statistics) show similar performances for the three one-class SVM formulations.

The preliminary experiments show that the brute-force attacks are easily detected. For example, flooding with 1000 INVITE/s in less than one second causes from 12 to 23 abnormal cycles (the cycle is of 5 seconds) depending on the request URI. The average results over multiple runs are depicted in Table II.

The real challenge is thus detecting the stealthy attacks (e.g. flooding with 1 INVITE/s). We experiment with different preset volumes of the background traffic (in

Table II
RESULTS FOR BRUTE-FORCE ATTACKS

Attack URI	Number of caused abnormal slots
Invalid user-name	7
Valid user-name	13
Valid user-name at another domain	12
Invalid domain name	23
Invalid domain IP address	12

calls/hour*bot). We train with the normal traffic during 100 minutes than we divide the samples in two equal subsets: one for training and one for testing. In addition, the testing subset regroups three kinds of anomaly (each anomaly has a duration of 5 minutes):

- Flooding using Inviteflood with a valid target and controlled intensity,
- SPIT using the range_spit functionality of one of the VoIP bots with a controlled intensity,
- Asterisk shut-down: In contrast to the two previous anomalies, this situation is used to test the detection of abnormal shortage of the SIP activities.

The results using the RBF kernel and the hypersphere formulation are depicted through the Receiver Operating Characteristic (ROC) curves in Figure 4. The ROC curves are obtained by varying the decision value of the decision function (i.e. varying the radius of the enclosing sphere). In the ROC plots, the X-axis represents the false positives that is the normal time cycles that are detected as abnormal. The Y-axis represents the true positives that is the abnormal time cycles that are correctly detected as abnormal. The best results hence corresponds to the top-left corner. The accuracy is measured as the Area Under Curve (AUC) of the ROC curve within the interval [0:0.1] since higher false positives are not tolerable in intrusion detection systems. As shown, the accuracy decreases in function of the ratio of the anomaly intensity to the background traffic volume. In conclusion, a high true positive rate (over 95%) is ensured under an acceptable false positive rate (below 3%) if the anomalous traffic is at least the double of the background traffic. The effect of the false positives can be mitigated by introducing an aggregation rule (e.g. an alarm is issued only if more than 5 successive positives are detected, that is within 25 seconds delay since the anomaly startup).

We also investigate the most relevant features for the prediction using the prediction sensitivity approach. The prediction sensitivity consists on adding weight to one feature and studying how the values returned by the decision function evolve. The ”salient” features are characterized by a high mean value for the anomaly observations while a large standard deviation corresponds to highly heterogeneous data in the normal set. We applied this technique to the three anomalies in question. The results using the sphere formulation and the RBF kernel are depicted in Figure 5. Not surprisingly, the top-ranked three features for flooding

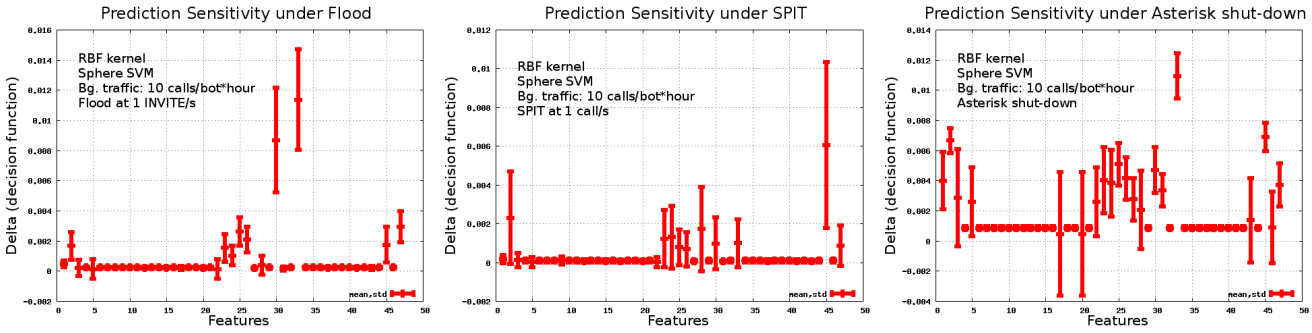

Figure 5. Prediction sensitivity for the three types of anomaly

detection are: the number of in-use transactions (feature n.33), the number of 4xx transactions (feature n.30) and the number of failed dialogs (feature n.49). Those for SPIT detection are the number of active dialogs (feature n.45), the number of received replies (feature n.2) and the number of 2xx transactions (feature n.28). The detection of the Asterisk shut-down (shortage of a part of traffic) is particularly sensitive to the number of in-use transactions (feature n.33), the number of active dialogs (feature n.45) and the number of received replies (feature n.2).

The approach shows real time proficiency. We provide Python scripts for online monitoring of Opensips with two modes for training and testing based on the Libsvm library[5] [6]. The performance of this tool is inherited from the Opensips performance and that of the LibSVM tool. Knowing that LibSVM implements an algorithm for sparse data (very high dimensionality) performances can be easily enhanced by using an algorithm for dense data.

VI. Related Works

Securing VoIP/SIP impose several research challenges that have been under debate in the research community. A complete VoIP security and privacy threat taxonomy has been published by VoIPSA (VoIP Security Alliance) in [7]. Geneiatakis et. al. analyzed the SIP security mechanisms that are currently deployed and highlighted their weaknesses [8]. Abdelnur et. al. developed new methodologies for VoIP networks assessment and discovering of vulnerabilities in SIP implementations [9].

Traffic monitoring and intrusion detection are especially challenging because of the open, dynamic and heterogeneous aspects of the SIP networks. Sengar et. al. [10] proposed specification-based detection based on the interaction between extended finite state machines of the VoIP protocols. By tracking deviations from interacting protocol state machines, this solution is able to detect a series of SIP, RTP and cross-protocol attacks. Similarly, Ding et al. [11] proposed a timed Hierarchical Colored Petri Net (timed HCPN) model to detect albeit the same set of attacks. Kang et. al. proposed a general methodology for profiling SIP traffic at three levels: the server host, the functional entities (registrar, proxy) and the individual user [12]. Their methodology which is entropy-based proved good efficiency for anomaly detection. Sengar et. al. [13] addressed the flood detection problem and proposed using the Hellinger Distance in order to measure the distance between training and testing probability distributions. More than 20 works on security systems to counter denial of service attacks have been discussed by Ehlert et. al. in [14]. Gritzalis et. al. addressed the SPIT problem and examined the state-of-the-art SPIT management techniques [15].

Our approach is not focused in scope towards one type of attack or relying on only one kind of data sources. We proposed an overall monitoring scheme for anomaly detection using SVM-based techniques. In [16], we defined 38 statistics (or probes) characterizing the SIP traffic and showed their efficiency for detecting a series of network traffic anomalies namely flooding and SPIT. In this paper, we continued this work and proposed an overall monitoring scheme by adding the server statistics and the call detail records to our model. Similar to our work is [17]: Yu et. al. proposed a hierarchical two-level SVDD (Support Vector Data Description) in order to classify flooding attacks (TCP-SYN, UDP and ICMP) using SNMP MIB statistical data.

VII. Conclusion

This paper sketched a global scheme for security monitoring in SIP networks. We have defined a set of relevant features for profiling and anomaly detection at three important levels: The SIP traffic, the server statistics and the billing records. We have proposed SVM techniques for anomaly detection and showed their accuracy and real-time efficiency. We have compared three one-class SVM formulations and proposed methods for anomaly and anomaly source identification. The experiments have been performed on a controlled test-bed where normal and attack conditions are synthesized.

In the future, we aim to extend this framework at both the traffic generation model and the anomaly detection model. The usability in real world settings is one of the

[5]http://www.loria.fr/~nassar/opensips_online_monitoring.zip

requirements that was not covered in this paper. In particular, the adaptive learning and the unsupervised learning issues must be considered. We will also study the correlation between the three monitoring levels. Finally, we aim to cover other anomalies and threats especially in the area of fraud detection.

REFERENCES

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "RFC3261: SIP: Session initiation protocol," United States, 2002.

[2] V. Vapnik, *Statistical Learning Theory*. New York: John Wiley and Sons, 1998.

[3] P. Laskov, C. Schfer, and I. Kotenko, "Intrusion detection in unlabeled data with quarter-sphere support vector machines," in *Proceedings of the first Conference on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA)*, 2004, pp. 71–82.

[4] P. Laskov, K. Rieck, C. Schäfer, and K.-R. Müller, "Visualization of anomaly detection using prediction sensitivity," in *Sicherheit*, 2005, pp. 197–208.

[5] "The VoIP Bot project," http://gforge.inria.fr/projects/voipbot/.

[6] C. Chang and C. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[7] VoIPSA, "VoIP security and privacy threat taxonomy," Public Realease 1.0, Oct 2005, http://www.voipsa.org/Activities/VOIPSA_Threat_Taxonomy_0.1.pdf.

[8] D. Geneiatakis, G. Kambourakis, T. Dagiuklas, C. Lambrinoudakis, and S. Gritzalis, "SIP security mechanisms: A state-of-the-art review," in *Proceedings of the Fifth International Network Conference (INC 2005)*, Samos, Greece, July 2005, pp. 147–155.

[9] H. Abdelnur, R. State, and O. Festor, "KiF: a stateful SIP fuzzer," in *Proceedings of the 1st international conference on Principles, systems and applications of IP telecommunications (IPTComm '07)*. New York, NY, USA: ACM, 2007, pp. 47–56.

[10] H. Sengar, D. Wijesekera, H. Wang, and S. Jajodia, "VoIP intrusion detection through interacting protocol state machines," in *Proceedings of the 38th IEEE International Conference on Dependable Systems and Networks (DSN'2006)*. IEEE Computer Society, 2006.

[11] Y. Ding and G. Su, "Intrusion detection system for signal based SIP attacks through timed HCPN," in *Proceedings of the Second International Conference on Availability, Reliability and Security (ARES'07)*. IEEE Computer Society, 2007.

[12] H. Kang, Z. Zhang, S. Ranjan, and A. Nucci, "SIP-based VoIP traffic behavior profiling and its applications," in *Proceedings of the 3rd annual ACM workshop on Mining network data (MineNet '07)*. New York, NY, USA: ACM, 2007, pp. 39–44.

[13] H. Sengar, H. Wang, D. Wijesekera, and S. Jajodia, "Detecting VoIP floods using the Hellinger distance," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, no. 6, pp. 794–805, 2008.

[14] S. Ehlert, D. Geneiatakis, and T. Magedanz, "Survey of network security systems to counter SIP-based denial-of-service attacks," *Computers & Security*, vol. 29, no. 2, pp. 225 – 243, 2010.

[15] D. Gritzalis and Y. Mallios, "A SIP-oriented SPIT Management Framework," *Computers & Security*, vol. 27, no. 5-6, pp. 136 – 153, 2008.

[16] M. Nassar, R. State, and O. Festor, "Monitoring SIP traffic using support vector machines," in *Proceedings of the 11th International Symposium on Recent Advances in Intrusion Detection (RAID '08)*. London, UK: Springer-Verlag, 2008, pp. 311–330.

[17] J. Yu, H. Lee, M. Kim, and D. Park, "Traffic flooding attack detection with SNMP MIB using SVM," *Computer Communications*, vol. 31, no. 17, pp. 4212 – 4219, 2008.